US009182997B2

(12) **United States Patent**
Sandstrom

(10) **Patent No.: US 9,182,997 B2**
(45) **Date of Patent: Nov. 10, 2015**

(54) **DIRECT BINARY FILE TRANSFER BASED NETWORK MANAGEMENT SYSTEM FREE OF MESSAGING, COMMANDS AND DATA FORMAT CONVERSIONS**

(71) Applicant: **Mark Henrik Sandstrom**, Jersey City, NJ (US)

(72) Inventor: **Mark Henrik Sandstrom**, Jersey City, NJ (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 16 days.

(21) Appl. No.: **14/038,685**

(22) Filed: **Sep. 26, 2013**

(65) **Prior Publication Data**

US 2014/0032891 A1 Jan. 30, 2014

Related U.S. Application Data

(63) Continuation of application No. 11/566,178, filed on Dec. 1, 2006, now abandoned.

(60) Provisional application No. 60/866,208, filed on Nov. 16, 2006.

(51) **Int. Cl.**

G06F 15/16	(2006.01)
G06F 15/173	(2006.01)
G06F 9/34	(2006.01)
G06F 9/44	(2006.01)
H04L 12/24	(2006.01)

(52) **U.S. Cl.**

CPC ***G06F 9/4416*** (2013.01); ***H04L 12/24*** (2013.01); ***H04L 41/00*** (2013.01); ***H04L 41/0846*** (2013.01); ***H04L 41/0856*** (2013.01); *H04L 41/22* (2013.01)

(58) **Field of Classification Search**

None

See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,566,335	A	10/1996	Nash et al.	
8,495,244	B2 *	7/2013	Bonar et al.	709/239
2004/0139308	A1 *	7/2004	Foster et al.	713/1
2006/0029085	A1 *	2/2006	Booman et al.	370/401
2006/0155825	A1 *	7/2006	Torii	709/217
2006/0161674	A1 *	7/2006	Sun et al.	709/230
2007/0083668	A1 *	4/2007	Kelsey et al.	709/238

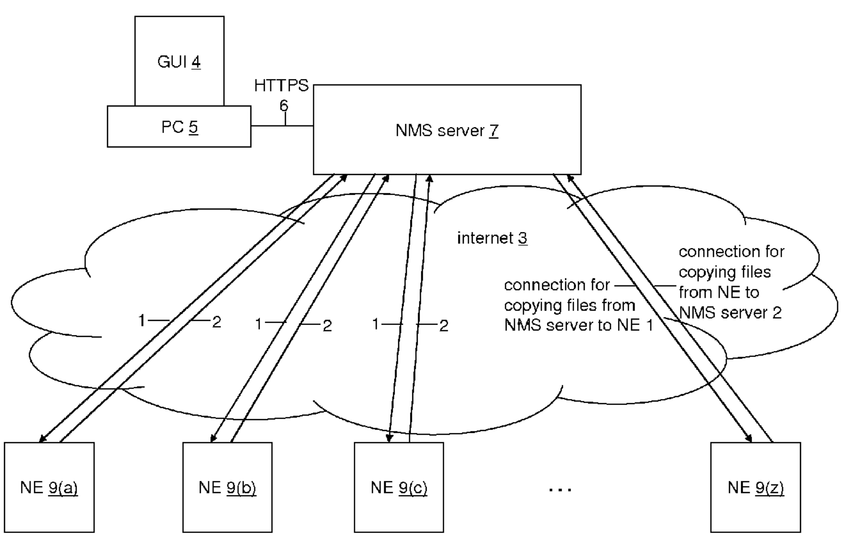
* cited by examiner

Primary Examiner — Ninos Donabed
Assistant Examiner — Farrukh Hussain

(57) **ABSTRACT**

Telecommunication network management operations are performed based on accessing network management data (NMD) files via GUIs and general purpose computers including a network management system (NMS) server, and automatic routines for transferring binary NMD files between the general purpose computers and remote network elements (NEs) being managed. A system user produces configuration files at the NMS server for NEs using a network management GUI, and the hardware of NEs automatically complete the network management operations indicated by the NMD files transferred to them from the NMS server and produce their status files to the NMS server. The network management GUI displays network status based on the latest NE status files at the NMS server. This provides direct, binary file transfer based NMS communication that avoids the complexity and restrictions of intermediate messaging protocols or transaction languages and conversions thereof.

16 Claims, 3 Drawing Sheets



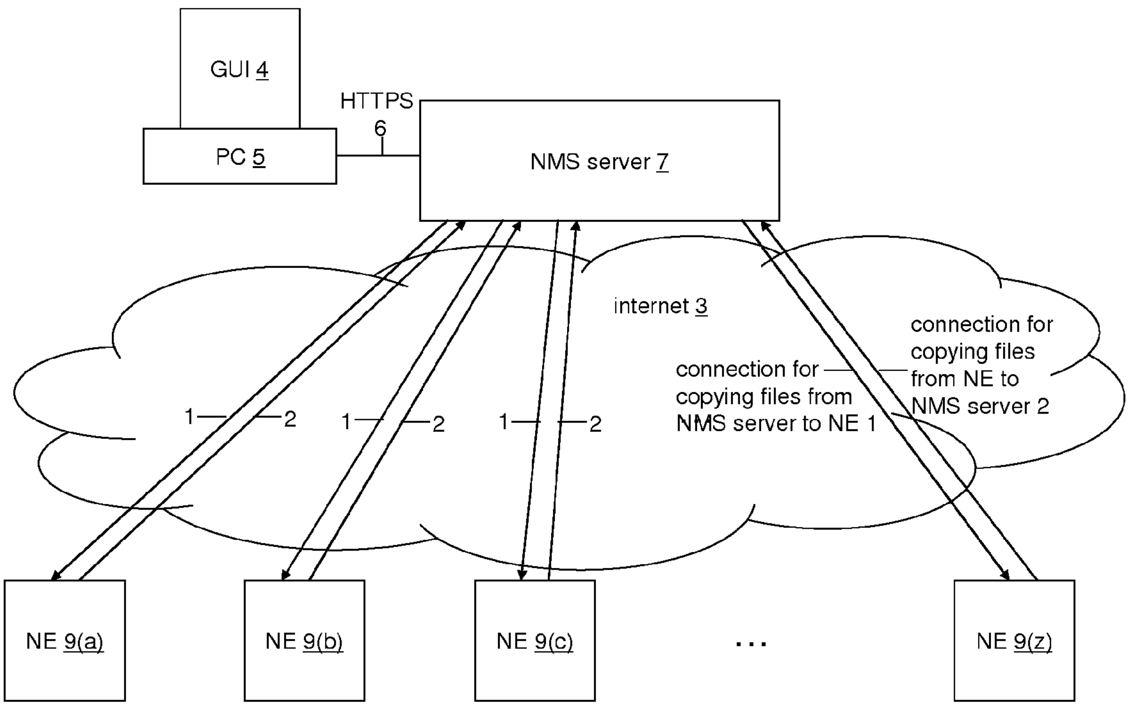

GUI 4
PC 5
HTTPS
6
NMS server 7
internet 3
connection for
copying files from
NMS server to NE 1
connection for
copying files
from NE to
NMS server 2
1
2
1
2
1
2
NE 9(a)
NE 9(b)
NE 9(c)
...
NE 9(z)

FIG. 1

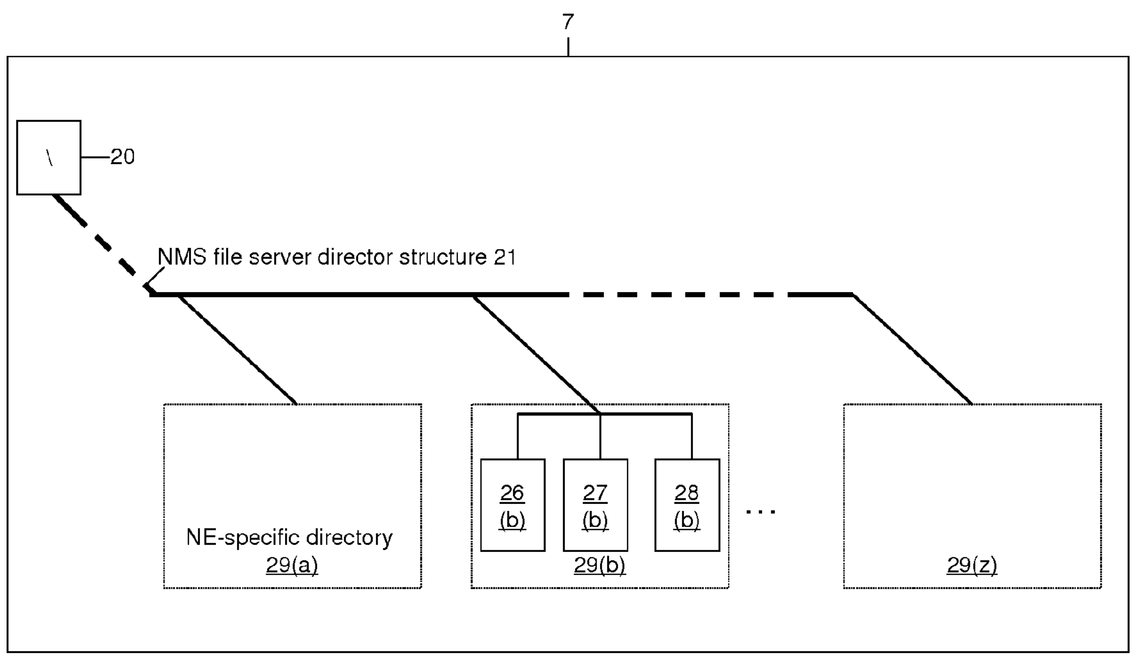

7
\
20
NMS file server director structure 21
NE-specific directory
29(a)
26
(b)
27
(b)
28
(b)
29(b)
...
29(z)

FIG. 2

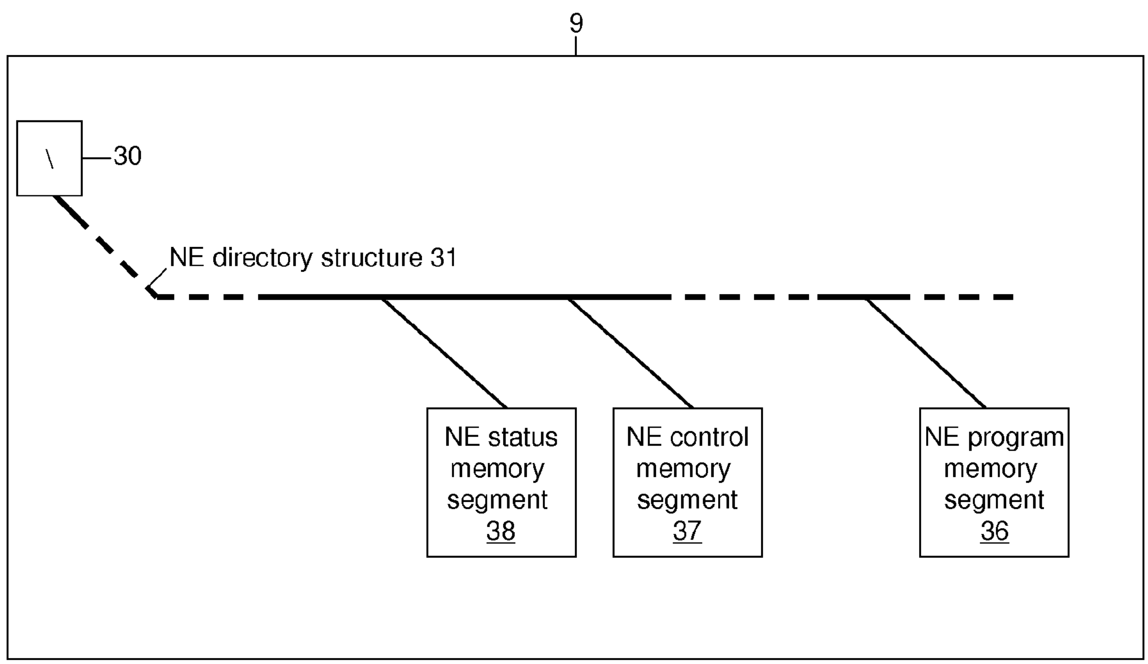

9
\
30
NE directory structure 31
NE status memory segment 38
NE control memory segment 37
NE program memory segment 36

FIG. 3

DIRECT BINARY FILE TRANSFER BASED NETWORK MANAGEMENT SYSTEM FREE OF MESSAGING, COMMANDS AND DATA FORMAT CONVERSIONS

CROSS REFERENCE TO RELATED APPLICATIONS

This application is a continuation of a U.S. application Ser. No. 11/566,178, filed Dec. 1, 2006, which is incorporated by reference in its entirety and which claims the benefit of U.S. Provisional Application No. 60/866,208, filed Nov. 16, 2006, which is incorporated by reference in its entirety (and referred to herein with the reference number [5]).

This application is also related to the following, each of which is incorporated by reference in its entirety: [1] U.S. application Ser. No. 10/170,260, filed Jun. 13, 2002, entitled "Input-Controllable Dynamic Cross-Connect"; [2] U.S. application Ser. No. 10/192,118, filed Jul. 11, 2002, entitled "Transparent, Look-Up-Free Packet Forwarding Method for Optimizing Global Network Throughput Based on Real-Time Route Status"; [3] U.S. application Ser. No. 10/382,729, filed Mar. 7, 2003, entitled "Byte-Timeslot-Synchronous, Dynamically Switched Multi-Source-Node Data Transport Bus System"; [4] U.S. application Ser. No. 11/245,974, filed Oct. 11, 2005, entitled "Automated, Transparent System for Remotely Configuring, Controlling and Monitoring Network Elements"; and [6] U.S. application Ser. No. 11/563,079, filed Nov. 24, 2006, entitled "Intelligent Network Alarm Status Monitoring."

BACKGROUND

The present invention pertains to the field of telecom network management systems, in particular to network management communications.

Acronyms used in this specification are defined below:

GUI Graphical User Interface
HW Hardware
IF Interface
NE Network Element
NMS Network Management System
PC Personal Computer
SW Software

Conventional telecom network management systems (NMS) rely on command and messaging based communications for transactions and information distribution between various elements of the NMS and the network being managed. Examples of such commonly used messaging and command based NMS communications techniques are various versions of Simple Network Management Protocol (SNMP), Transaction Language 1 (TL1) and Common Management Information Protocol (CMIP). Moreover, conventional NMS implementations are often based on complex, technology and system vendor specific concepts, such as Management Information Bases (MIBs), for storing or representing various sets of network management objects, and various types of methods to operate on, e.g., given MIB objects.

These conventional NMS transactions and methods are normally event-triggered such that they can occur spontaneously or in an unsolicited manner. For example, they may occur based on dynamic events that take place on network data plane, based on automated NMS response to particular information received from the network being managed, or based on human operator initiated network management operations. For instance, network defect or fault activations and de-activations cause NMS messages among various elements of NMS implementations with conventional NMS technologies. Likewise, individual transactions, such as accessing a given parameter at a network element (NE) (e.g., reading a NE performance monitoring status register or re-configuring a NE control register), involve their specialized messaging and command based transactions with conventional NMS techniques. Moreover, to complete even such a basic NMS transaction, several stages of protocol, message, language and data format conversions are involved with conventional NMS implementations. These messaging based prior art NMS schemes are prone to become overloaded during times of high loads of NMS and network event activities. As a consequence, conventional network management and NMS communications systems and methods are reactive in their nature and impulsive in operation, causing several significant problems inherent with them. These problems include the following:

- The conventional NMS performance degrades when the NMS capabilities are most urgently needed, e.g., during bursts of messaging and transaction triggering network events, e.g., major network failures.
- Several functional components of conventional NMS techniques are vendor dependent or vendor specific. This causes the full system integration to become complicated and requiring various stages of integration SW, i.e., middleware to be designed between the functional components, resulting in lost NMS transparency from NMS user IF to the network element HW, reduced system flexibility and increased cost.
- Many conventional NMS techniques are specific to certain protocols, languages or data formats, causing the need for various stages of protocol conversion agents and the like, further complicating the conventional NMS implementations while making them less transparent and less flexible.
- Since NMS operations with conventional systems and technologies are typically based on a predefined, fixed set of commands or methods (e.g., requests, responses etc. predefined atomic transactions) specific to and limited by the technologies in use at a given implementation, the scope of possible functionality supported through conventional NMS are commonly strictly restricted to only such a subset of capabilities of the components of the NMS implementation that is supported by each component throughout the implementation.

It is thus observed that, even with their exhaustive implementational complexity, conventional NMS techniques are usually inefficient in operation. Accordingly, there is a need for innovations that enable streamlined network management communications, providing more intuitive, transparent and flexible operational capabilities with architecturally improved scalability, reliability and performance, especially under heavy load of network management and network data plane event activities.

SUMMARY

Embodiments of the invention provide efficient systems and methods for unrestricted network management operations based on the transfer of binary network management data (NMD) files between a network management systems (NMS) server and remote network elements (NEs) being managed through the NMS.

In one embodiment, a method for managing NEs includes repeatedly, via periodical routines, transferring NE program and control files from NE-specific directories at the NMS server to their related NEs, and transferring NE status files from NEs to their associated directories at the NMS server. The NE control files represent the intended binary contents of their corresponding NE control register segments, and the NE status files represent the binary contents of NE status register segments within the memory space of their associated NEs.

In one embodiment, a network management system for configuring and monitoring remote NEs comprises an NMS file server for storing binary NMD files associated with the NEs. The system is configured to periodically transfer NMD files between the file server and the NEs. A graphical user interface (GUI) provides user access to the NMD files at the NMS file server. The intended actions associated with the NMD files transferred to the NEs generally occur as automatic consequences of the NEs storing these files at their local memories. The NMS GUI automatically displays current network status based on the NMD files transferred from the NEs to NMS file server. The NMS file server provides dedicated directories for NMS files of each NEs being managed through the NMS, and the NEs know which NMD file directories at the NMS server to access based on their unique NE IDs configured individually for each NE hardware unit, as the NE-specific NMD directory names at the NMS server, in an embodiment of the invention, include the NE ID for the NE associated with each directory.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1** illustrates the functional architecture for a network management communications system involving an NMS file server and a set of NEs, in accordance with an embodiment of the invention.

FIG. **2** is a diagram of a logical directory structure at the NMS file server shown in FIG. **1**.

FIG. **3** is a diagram of a logical structure of a local memory space of an NE shown in FIG. **1**

The following symbols and notations used in the drawings:

- A box drawn with a dotted line indicates that the set of objects inside the box form an object of higher abstraction level, such as in FIG. **2** an NE specific directory **29** comprising sub-directories **26**, **27** and **28**.
- Lines or arrows crossing in the drawings are decoupled unless otherwise marked.
- Arrows between boxes in the drawings represent a path of information flow and can be implemented by any communications means available, such as Internet or Local Area Network based connections. A line connecting elements is considered a bi-directional communication path unless a direction is indicated with an arrow.
- The symbol '\' represents a logical root of a file system or a directory structure.
- Three dots between instances of a given object indicate an arbitrary number of instances of such an object, e.g. NEs **9** in FIG. **1**, repeated between the drawn instances.

The figures depict various embodiments of the present invention for purposes of illustration only. One skilled in the art will readily recognize from the following discussion that alternative embodiments of the structures and methods illustrated herein may be employed without departing from the principles of the invention described herein.

DETAILED DESCRIPTION

FIG. **1** presents an overview of functional architectural of the network management process of present invention. At a high-level, the NMS of the invention, via a set of automatic routines, transfers binary network management data (NMD) files between an NMS file server **7** and a set of NEs **9**(*a*), **9**(*b*), **9**(*c*) etc. (later referred simply to as NEs **9**, or NE **9** for any individual one of the NEs **9**(*a*) through **9**(*z*)), while network management actions occur as automatic consequences of the contents of the NMD files.

The NMS process of FIG. **1** is based on the following mutually asynchronous and conceptually de-coupled sub-processes:

1) A set of automatic file transfer routines transfers NMD files between the NMS server **7** and the NEs **9**;

2) The NEs perform on their end the appropriate NMS actions associated with the NMD files;

3) The NMS GUI **4** acts on the NMD files at the NMS server **7**, to perform the NMS transactions on its end.

The sub-process **1**) in one embodiment is based on a secure Network File System (NFS), with the NMS server **7** providing NFS server and the NEs **9** NFS client functionalities. This sub-process further comprises the below two NMD file transfer routines that the NEs repeat periodically, e.g. every 1, 5 or 10 seconds:

a) The NFS clients of the NEs **9** look for and copy **1** their associated NMD program and control files, referred to as NE configuration files, from directories at the NFS server **7** designated for configuration files intended for their respective destination NEs, over a network **3** to the local memories at the NEs. A NE looks for its new program and control files at their respective, predefined directory locations designated to that NE at the NMS server, and after copying such files from the NMS server, stores its program files at a program memory segment, and its control file at a control register segment within its local memory space.

b) In addition, the NFS clients at the NEs **9** copy **2** contents of their status register segments within their local memories via a type of NMD file referred to as NE status file over a network **3** to their associated directories at the NMS server **7** designated to status files from the individual NEs managed through the NMS.

In one embodiment, the above routines of a) transferring NE configuration files from the NMS server to NEs, and b) transferring NE status files from NEs to the NMS server are independent operations, i.e., neither one either triggers the other or is caused by the other.

The sub-process **2**) in one embodiment is performed by the NE HW, e.g. per referenced application [5], automatically based on the binary contents of the NE control files, normally without further involvement by either NMS or NE SW. An exception to that is a case when a NE control file contents contain such a value in a particular NE control register, referred to as reboot control register, that is intended to cause the NE SW to reboot, in which case the NE SW will do a reboot of a type indicated by the reboot control register value. Aside this reboot exception, i.e., in cases when the reboot control register in a NE control file does not indicate a reboot action, the NE hardware automatically, without SW involvement, completes the network management actions indicated by the contents of new NE control files copied **1** to the control register segment in its local memory space. The NE also copies to its program memory segment within its local memory space any new program files from its associated directory at the NMS file server designated for program files for that destination NE. In one embodiment, the program memory segment of a NE comprises multiple directories to allow storing multiple NE program files, and the value of the NE reboot control register indicates both whether the NE is to reboot, and using program files in which directory in the program memory segment. In addition, the NE HW automatically maintains and updates a set of NE status parameters in its status register memory space, and the NE SW reads the contents of this status register segment in the NE memory space to a NE status file that the NE copies **2** to an appropriate directory at the NMS server designated for status files from that source NE. In one embodiment, the NE copies **2** also the contents of its control registers via its NE status file back to the NMS server, allowing the user to verify the actual values of also the NE control registers via GUI **4**. Hence, the phrase status NE status file herein shall include the contents of both the NE control and status registers, collectively referred to as NE device registers.

The sub-process **3**) in one embodiment is performed by the NMS GUI SW **4** via providing access in a human understood format for the system user to the NMD files at the NE-specific directories at the NMS file server **7**. This sub-process involves write and read access to control register values within the NE control files via the NMS GUI, producing NE program files to appropriate folders at the file server **7**, and read access to the NE status files at the server **7**. Moreover, in one embodiment the NMS GUI **4** displays notifications of significant events in the network such as NE alarm activations according to principles per referenced applications [5] and [6].

It is seen from the above discussion that the three main sub-processes of the NMS process of the invention are mutually de-coupled, other than through the contents of the NMD files that indicate the intended actions to be performed by other elements of the system to complete any given network management operation. Compared against prior art messaging and command based NMS techniques, this de-coupling between the functional elements of the network management system and methods of the present invention yields several benefits over the, including the below ones:

- Heavy load of NMS or network event activity on one element of the system per the invention does not negatively interfere with other elements. For instance, while e.g. the NMS server **7** is heavily loaded during for instance a network service contract testing period when the NE control parameters are changed rapidly for test case purposes, the file transfer routine, the SW of the NEs, and even the HW of those NEs not under the test, are not at all impacted. Likewise, a heavy load of e.g. network defect activity at a given NE does not impact the NMD file transfer routines, the other NEs, the NMS server or GUI SW; instead, in one embodiment, e.g. per the referenced applications [5] and [6], just a single NE alarm notification is generated at the NMS GUI when a previously defect-free NE enters a defected state. As a consequence, the NMS per the invention is highly reliable and scalable, providing a predictable, steady performance under any load of NMS and network event activities.
- The system per the invention is flexible regarding any changes needed to the implementation of either the NE **9**, NMS server **7**, GUI **4** etc. elements of it, as well as any changes to the network **3** through which the NMS and NEs transfer files, or to the way the GUI and the NMS file server communicate **6**. Consequently, any of these system elements can change without having to redesign the rest of the NMS system.
- The system per the invention provides transparent NMS communications all the way from the NMS GUI **4** to the NE **9** hardware device registers and back, without intermediate messaging protocol conversion or command translation agents etc. non-transparent middleware common with prior art NMS communications techniques. Accordingly, the invention inherently enables a more intuitive and flexible network management, by allowing direct access to the NE parameters of interest via an intuitive and transparent GUI, without requiring the network operator's personnel to know about or deal with the peculiarities of any intervening messaging protocol or command language syntax.

A possible implementation of the NMS of the invention further comprises a PC **5** hosting the NMS GUI application, e.g. HTML based web browser **4**. In such a system implementation, the GUI **4** connects to the NMS server **7** over a secure HTTP connection **6**. Regarding FIG. **1**, it shall be understood that there is no implied limit to the number of NEs supported by this network alarm monitoring system, but that instead this system architecture supports an arbitrary number of NEs **9**, and that there can equally well be multiple physical NMS server **7** and user computers **5** as well as multiple concurrent NMS GUI applications **4**.

FIG. **2** illustrates a logical directory structure according to the invention at the NMS file server **7** for storing NMD files for a set of NEs managed through the NMS. Below the file server directory root **20**, there is a directory structure **21** holding a set of NE-specific directories **29**(*a*) through **29**(*z*) (later simply directories **29** for the set, or a directory **29** when referring to any one of the similarly structured directories **29**(*a*), **29**(*b*), **29**(*c*) etc.). Each of the directories **29** stores NMD files for its associated NE **9**, for example directory **29**(*b*) at the NMS server **7** stores NMD files associated with NE **9**(*b*) (FIG. **1**). In one embodiment, each directory **29** comprises subdirectories **26**, **27** and **28** for holding program, control and status files, respectively, of the NE associated with the directory. These subdirectories have been drawn in FIG. **2**, as an example, for the case of directory **29**(*b*) associated with NE **9**(*b*) (FIG. **1**). Each of the NE-specific directories **29**, however, have similar subdirectory structure as the directory **29**(*b*).

Operation of the NMS file server **7** in a process of configuring and monitoring a given NE **9** in one embodiment is based on the below principles:

1) A system user, e.g., a network operator staff member, produces desired types of NE program and control files for a NE **9**, using the NMS GUI client **4** and related server software at the NMS files server **7**, into the program file directory **26** and control file directory **27** associated with the NE **9**.

2) The NE **9**, via a repeating routine, for instance every fifteen seconds, looks for and copies these files from its associated directories **26** and **27** at its NMD directory **29** at the NMS server to their appropriate locations within the local memory space of the NE. The NE will consequently autonomously complete on its end the NMS operations indicated via each new NE configuration file.

3) The NE **9**, also via a repeating routine performed e.g. once every second, copies the contents of its device status registers via its NE status file to the folder **28** at its directory **29** at the NMS server **7**. The NMS SW will consequently display NE status data, along with a new NE alarm notification as necessary, to the user via the GUI **4**, based on the contents of the latest NE status file at its associated directory **28** at the NMS server **7**, in one embodiment utilizing the network alarm monitoring principles per the referenced applications [5] and [6].

The management process for a group of NEs, e.g. **9**(*a*), **9**(*b*) and **9**(*c*) in FIG. **1**, is based on simply repeating, or executing in parallel, the management process of a single NE described above. Copying of files between directories at the NMS server **7** and the NEs **9** in one embodiment is based on secure NFS, specifically NFS version 4 in one embodiment. Also, in one embodiment, each NE hardware unit **9** is identified by its unique NE ID number configured at the factory on a nonvolatile memory, e.g. flash drive, for each NE unit **9**, and the names of the NE-specific directories **29**(*a*) through **29**(*z*) at the file server **7** include the NE ID of their related NEs, based on which each NE **9** knows to access its appropriate directory **29** at the NMS server **7**.

It is observed that a need for identifiers for source, destination, message or transaction is avoided with the present invention for NMS communications between the NMS server **7** and the NEs **9**, in part via the use of NE-specific directories at the NMS file server **7** for storing the NMD files associated with each one of NEs **9**(*a*) through **9**(*z*) (FIG. **1**). Note that transaction, source, destination etc. identifiers are usually necessary with prior art NMS communications schemes, per the prior art messaging protocols (e.g. SNMP, CMIP, TL1 etc.), requiring related NMS messaging protocol processing to be performed by prior art NMS and NE devices, thus making the prior art network management systems implementation, operation and administration more complicated and less flexible compared to the plain binary NMD file transfer based NMS communication of the present invention. Besides its more straightforward and transparent implementation and more flexible and intuitive operation, benefits of the invention enabled via the NE-specific NMD file folders include elimination of NMS communications overhead that is needed with prior art system for their NMS messaging protocols, and the clarity and intuitiveness of the NMS file server structure based on a repeated set of similar NE specific directories **29**(*a*) through **29**(*z*) for the set of NMD files **26**, **26** and **28** per each NE **9**(*a*) through **9**(*z*).

It shall be understood that the term directory herein refers to a file folder i.e. directory at any level of the file system hierarchy, and that e.g. phrase NE-specific directory can be used to denote a hierarchical directory with sub-directories for holding NMD files of a given NE, as well as a bottom level directory used to hold a single NMD file of a particular type. Also, it shall be understood that there can be any desired number of NE specific directories **29** at the NMS server **7**, that there can be any desirable number of levels of file system hierarchy within directory structures **21**, **26**, **27**, **28** and **29**, and that the directory structure **21**, as well as the sub-structures of directories **29**, can include other directories and files in addition to the ones shown in FIG. **2**.

FIG. **3** illustrates logical structure of local memory space of each NE **9** of the network management system of the invention (FIG. **1**). The embedded memory space of the NE **9** comprises a program memory segment **36**, a control register segment **37** and a status register segment **38**. It shall be understood that various embodiments of NEs **9** can have various other memory segments, e.g., RAM, in addition to the three segments shown in FIG. **3**, and that there can be as many layers of hierarchy of NE logical directory structure **31** below its root **30** as desired in each embodiment, as well as that the shown memory segments **36**, **37** and **38** can have sub-directories. Reference specifications for one embodiment of a NE, including NE device register descriptions with related application notes, are provided in the referenced application [5].

In one embodiment, the NE memory space is organized as a logical directory structure **31**, with the NE program memory segment **36** forming a logical subdirectory at the NE for holding the NE program files, and the NE control register segment **37** and the NE status register segment **38** each forming binary files under the NE logical directory structure **31**. The NE **9** may comprise a HW unit with an embedded microprocessor and a set of embedded memories organized from the NE SW perspective as a continuous directory structure. In one NE HW embodiment, the NE program memory directory **36** is a flash drive, and the NE control and status files are predefined address ranges within the embedded memory space of the NE microprocessor containing the NE device control and status registers, respectively. Furthermore, in one embodiment, the NE device registers are implemented within a digital logic device that is configured, at least in part via the NE program files stored at the directory **36**. In such an embodiment, NE digital logic device completes on the NE side the network management operations indicated via each new NE control file **37**, as well as produces and keeps updated a predefined set of NE status parameters on the NE status file **38**. Such NEs capable of operating autonomously and dynamically with this type of NMD files, including with a NE program files **36** and control file **37** that are static for a duration of a network service contract the NE is deployed for, in one embodiment are based on principles of referenced applications [1], [2], [3], [5] and [6].

Enabled by such intelligent, self-operating NE. HW. that in an embodiment utilize principles based on the referenced applications [1], [2] and [3], the NE SW does not need to perform dynamic responses to network events. Moreover, intelligent NEs may handle their NMS communications via periodically repeating routines for transferring binary network management data (NMD) files between an NMS server at service provider's NOC and their local memories, based e.g. on principles of referenced application [4], resulting in that the NE SW does not need to handle any NMS messaging, command etc. event alert, request or response type of transactions either, as all information distribution related to NMS communications, including alarm notifications, is carried via the binary NMD files representing NE device register contents, and as all NE-side network management actions (other than of course NE reboot actions) occur as automatic side-processes performed by the NE HW without a need for SW involvement other than copying NMD files between the local NE memories and the NMS server. I.e., the NE SW in an embodiment does not need to analyze or process the binary contents of the NMD files at all. Furthermore, since the NE HW, as described in referenced application [2], is able to forward Layer 2/3 (e.g. MPLS) packets without a need for any packet-layer switching, forwarding or routing tables, the intelligent NEs do not consequently need to maintain or sync any packet layer forwarding etc. tables either.

Accordingly, since the embodiment of the NE HW perform all the response-time critical processing at the NEs with sufficient HW logic resources available for all potential network events in any given network contract application that the NE is deployed for with network management configuration that is assumed static for the contract duration, while the NE SW executes periodic routines to carry out the NMS communications via the NMD file transfers between the NE and the NOC, the NE operation and performance is reliable and predictable under any loads of network data plane event and network management communications activities. I.e., the NE performance stays satisfactory regardless of e.g. its associated defect or alarm activity, data traffic patterns or load levels on its network interfaces, etc. events or any extreme conditions that can occur in its network service contract application. Based on intelligent NE HW capable of operating dynamically under static network management control, and on NMS communications based on repeated, transparent NMD file transfer routines that the NE SW are able to continuously handle with the NMS server, embodiments of the invention therefore provide a novel NE functional architecture of enabling predictable, high-quality network data plane as well as network management communications operation.

The Appendix A, together with the referenced applications [1], [2] and [3], provides specifications for a NE of an embodiment for applications such as MPLS and SDH/SONET protocol based telecom carrier networks. The Appendix A in particular provides reference specifications for device registers of an embodiment of a NE. As described in Appendix A, a NE HW may provide SDH/SONET circuit multiplexing, cross-connect and transport functionality with a dynamic switching capability (e.g., based on referenced application [1]), along with transparent, look-up-table free MPLS packet forwarding functionality (e.g., based on referenced application [2]) and dynamic network capacity allocation optimization functionality (e.g., based on referenced application [3]). Appendix A further details a set of NE control and status registers for network applications utilizing these dynamic SDH switching, look-up-free MPLS forwarding and dynamic bandwidth allocation optimization techniques. In an embodiment, NE control register contents are provided to the NE from the NMS server via a binary NMD file called a NE control file, and the contents of the NE device registers, including the NE status registers containing the NE alarm, defect status as well as performance monitoring parameters, are provided from the NEs to the NMS server via another binary NMD file called a NE status file. In one embodiment, the NMD file transfer between the NMS server and the NEs is based on a secure Network File System. e.g. NFS version 4.

In chapters 4.2 and 4.3 below, the device registers labeled as control registers form the active contents of the NE control file i.e. the populated address locations of within the NE control file, and similarly the device registers labeled as status registers form the active contents or the NE status file i.e. the occupied address locations within the NE status file. It is naturally also possible to take a copy of the NE control register address range, to form a read-only copy of the actual contents of NE control registers for e.g. NMS verification or reference use. It is thus possible that in a certain application, contents of the entire NE device register address range, including both the status and control registers, are copied to a NE status file to provided to NMS.

Within the status register charts of Ch. 4.3, it is seen that all NE elementary defect registers are logically OR:ed, through a hierarchy of intermediary alarm and defect resolution bit vectors, to form a NE top-level alarm indicator i.e. the im alarm bit. The NE defects register bits can also be suppressed by their related 'alarm en' control register bits, to prevent unmonitored or non-critical defects from causing a NE alarm.

It is observed that the NE HW does not need to provide interrupts for the SW, since HW interrupts are unnecessary with this NE functional architecture, since the NE HW is self-operating, and capable for re-configuring itself dynamically, even under static SW control, e.g. via the dynamic TS multiplexing in the XC (see Ch. 4.5) and network load and connectivity status sensitive MPLS packet layer protection and re-forward functionalities (see Ch. 4 6).

Also, programming notes are provided for the NE device control registers for an example application, in order to illustrate how a given contract application, in which the intelligent NE HW per the invention is able to operate dynamically based on the network data plane events, can be defined for the NE via a set of static NE control parameters i.e. the NE device control register contents. Via this example, it is seen that the appropriate NE control register values can be derived, in a preferred embodiment computed automatically by SW programs, directly from commercial contract definition parameters, namely in this example from the number of nodes i.e. IM NEs involved in a given network service contract.

Note however that the programming notes provided in the register description tables in Ch. 4.3 for the IM NE control registers are for a particular, practical example application, specifically for a symmetric, distributed MPLS switch over STM-16 ring, referred to as a symmetric contract. These programming notes provided for that example application however shall not be understood in a limiting sense, as the control registers can be programmed to various other values for various other applications.

4.3.2 Global Status

Register	mp_addr[7:0]	Bits	Description
im_alarrm	0	0	OR function of im alarm vctr (below, at address +1)
im_alarm_vctr	1		FM top-level alarm register, module alarms ANDed with their respective bits 04 im alarm en vctr.
sec_alarm		43	[sec_prior OR sec_secor OR (NOT sec_lock) OR sec holdover OR (NOT sec_155m_lock)] AND im_alarm_vctr_en[13]; check sec status in address +1.
abi_alarm		12	abi_alarm_status AND im_alarm_vctr_en[12]; check the ABI block status alarm 0
stm4_alarm_vctr		11:0	stm4_alarm_status_vctr ANDed with im_alarm_en_vctr[11:0], check the start address of the relevant STM-4 block. STM-4 #z of STM-16 #0 is represented by bit #z. STM-4 #z of STM-16 #1 is represented by bit #4+z. STM-4 #z Access is represented by bit #8+z, If bit #[b] is set, see mp_addr[10:6] = 4+b., with mp_addr[11]=1 and mp_addr[5:0]=0, I.e., read register ad address offset
sec_status	2		
sec_prior		4	Primary ref_clk_19m_out out-of-range indicator.
sec_secor		3	Secondary ref_clk_19m_out out-of-range indicator.
sec_lock		2	SEC is locked-in to reference indicator.
sec_holdover		1	SEC is in holdover mode indicator.
sec_155m_lock		0	Indicator of whether the jitter filter of 155 MHz SEC is locked to the SETS 19.44 MHz reference.
id	3	15:0	IM ID; bits 15:8 identify the IM FPGA via code FEh, and bits 7:0 identify the version nr (#1...127).

4.6.1.2 Status			
Register	mp_addr[7:0]	Bits	Description
abi_alarm_status	0		
vctr			
abd_w_alarm		3	Alive defect related to West RX AMB; read address +192 next.
abd_e_alarm		2	Active defect related to East FAX AMB; read address +128
ingr_pos_ w_ alarm		1	Active defect related to RX access PPP link; read address +32+1 next.
ingr_pos_e_alarm		0	Active defect related to RX access PPP link; read address +1
ingr_alarm_status	w+1		w=0 for East side, and w=32 for West side
dtim		3	dTIM
dplm		29	dPIM
drdi		1	dRDI; entered (cleared) when RDI of 1 (0) is received in three consecutive frames
dexc		0	dEXC
ac_ti	w+4		
ti_unstable		8	Active states indicates that RxTI has not been constant during the last four frames.
ac_ti		770	dUNEQ= (dTIM & AcTI = 0); dAIS = (dTIM & AcTI = FFh)
ac_sl	w+5		
sl_unstable		8	Active states indicates that RxSL has not been constant during the last four frames.
ac_s_I		7:0	
pn_ebc	w+6	7:0	pN EBC: one second. near-end erroneous block count (EBC), saturates at 255.
pf_ebc	w+7	7:0	pF_EBC; one second far-end EEC, saturates at 255.

4.6.2.2 Status			
Register	mp_addr[1:0]	Bits	Description
abm_fifo_pkt_disc_cnt	0		
abm_bulk_pkt_disc_cnt		15:8	Bulk packets dropped due to ABI ingress buffer overflow since this register was last read.
abm_prio_pkt_disc_cnt		1:0	Priority packets dropped due to ABI ingress buffer overflow since this register was last read.
refwd_fifo_pkt_disc_cnt	1		
refwd_bulk_pkt_disc_cnt		15:8	Bulk packets drooped due ABI re-forward buffer overflow since this reaister was last read
refwd_prio_pkt_disc_cnt		7:0	Priority packets dropped due to ABI re-forward buffer overflow since this register was last read.

A possible NE hardware implementation comprises, besides the embedded microprocessor and its memories, a programmable logic device (PLD, or FPGA i.e. Field Programmable Gate Array) within which, in one embodiment, both the microprocessor as well as the hardware logic, including the NE device registers, are included. With the programmable NE hardware, the NE program files in the flash drive **36** (as well as files in directory **26** in FIG. **2**) shall contain both a binary file for configuring the programmable hardware logic of the NE (its PLD/FPGA), as well as a binary executable program for the NE microprocessor. Various embodiments of NEs for the invention can naturally contain any number of microprocessors, logic devices as well as other hardware components.

The NE SW in one embodiment executes periodically, e.g. once every ten seconds, a repeating routine comprising the below steps:

1) The NFS client of the NE **9** looks for and copies to its local memory segments **36** and **37** new NE program and control files, respectively, at its associated directories **26** and **27** at the NFS server of the NMS server computer **7**;

2) The NE HW automatically completes the NMS actions indicated via new NE control files **37**;

3) The NE NFS client copies its status file **38** to its associated directory **28** at the NFS server **7**.

While the step 2) generally is performed by the NE HW, the NE SW in one embodiment however checks the value of a particular address in the control register segment referred to as the reboot control register after it has copied a new NE control file **37** from the NMS server. In case that the reboot control register was set to a value indicating NE reboot action, the NE SW will perform a type of NE reboot specified by the value of the reboot control register. However, as a general rule, i.e., in cases that the reboot control register was not set in an active value, the NE HW will complete all the network management operations indicated by each new NE control file **37** automatically without any SW involvement. Benefits of this NE functional architecture per the invention include that the NE SW does not need to process the NE control or status files **37** and **38** or perform related consecutive actions, other than checking the reboot command register in the NE control files and rebooting the NE as necessary. Consequently, the processing load for the NE SW is significantly reduced while the entire system operation is made faster, transparent and more predictable and reliable via HW automation, and elimination of intermediary processing stages with the NMS communications.

Regarding the NE memory space structure depicted in FIG. **3**, it shall be understood in various embodiments the NE control and status register segments **37** and **38** may consist of sub-segments in their respective address space ranges, that the device register segments **37** and **38** are not required to contain actual data storage elements for each of their bit and byte address locations, that any one or even all of the NE configuration file memory segments **36**, **37** and **38** can comprise multiple files or directories, and that any two or more of these memory segments **36**, **37** and **38** can logically be combined into a single file or directory.

Conclusions

This detailed description is a specification of various embodiments of the present invention. Specific architectural, system, process and logic implementation examples are provided in this and the referenced patent applications for the purpose of illustrating various embodiments and implementations of the invented concepts. Naturally, there are multiple alternative ways to implement or utilize, in whole or in part, the principles of the invention as set forth in the foregoing.

For instance, while the presentation of the network management system functional architecture of the invention (an overview of which is shown in FIG. **1**) is reduced to illustrating the organization its basic elements, it shall be understood that various implementations of that architecture can have any number of NEs served by an NMS server, any number of NMS servers, and any number of NMS GUIs, etc. Also, in different embodiments of the invention, the sequence of software and hardware logic processes involved with the network management communications system can be changed from the specific sequence described, and the process phases of the network management methods could be combined with others or further divided in to sub-steps, etc., without departing from the principles of the present invention. For instance, in an alternative embodiment, the NMS server could copy status files from the NEs and copy configuration files to the NEs, instead of NEs copying their status files to the NMS server and copying configuration files from the NMS server. Moreover, the logical functions that are described as implemented in hardware could in alternative implementations be implemented in software, and vice versa.

Generally, those skilled in the art will be able to develop different versions and various modifications of the described embodiments, which, although not necessarily each explicitly described herein, utilize the concepts of the present invention and are thus included within its spirit and scope. It is thus intended that the specification and drawings of this patent application considered not in a restrictive sense, but as exemplary only, with the scope of the invention being indicated by the following claims.

What is claimed is:

1. A network management process comprising:

transferring, via a set of repeating automatic file transfer routines, binary network element (NE) control files between a network management system (NMS) server and NEs, with the NE control files representing contents of segments within memory spaces of the NEs associated therewith, and whereby a given one of the NEs copies its respective NE control files from the NMS server to a control register segment within its memory spaces; and

performing, by the given NE, a set of NE-side actions for network management operations associated with a given one of its respective NE control files, with such NE-side actions performed i) based on binary contents of the given NE control file indicating a set of intended network management actions to be completed and ii) automatically by hardware without software involvement, at least when the contents of the given NE control file does not contain a value that causes software to reboot the NE;

wherein said NE-side actions for network management operations associated with the given NE control file comprise configuring, through contents of the control register segment within the NE memory spaces, how the NE hardware, automatically without software involvement, is to function, wherein said contents of the control register segment are direct copies of the binary contents of the given NE control file, and

wherein the NE-side actions performed automatically by hardware without software involvement include producing a set of NE status parameters, with such producing comprising: forming NE alarm indicators based at least in part on NE defect bits and alarm enable bits, wherein values of the alarm enable bits are based on the contents of the given NE control file, and wherein the set of NE status parameters include: a defect indicator, an intermediary level alarm indicator, or a top level alarm indicator for the NE.

2. The process of claim **1** wherein the given NE copies the given NE control file from its respective directory at the NMS server designated for control files for that given NE.

3. The process of claim **1** wherein the transferring further comprises: copying, by the given NE, its respective NE program file from the NMS server to a program memory segment within its memory spaces.

4. The process of claim **1** wherein the transferring, between the NMS server and the given NE, occurs such that the given NE is a separate and remote node in a network from the perspective of the NMS server.

5. The process of claim **1** wherein the NE-side actions performed automatically by hardware without software involvement include: connecting network traffic in an alternative manner, at least in part based on control bits associated with such connecting, wherein values of said control bits are based on the contents of the given NE control file.

6. The process of claim **5** wherein the connecting comprises: multiplexing, redirecting or re-forwarding.

7. The process of claim **1** wherein the NE-side actions performed automatically by hardware without software involvement include: handling network control information in one of alternative manners, at least in part based on control bits associated with such handling, wherein values of said control bits are based on the contents of the given NE control file.

8. The process of claim **7** wherein:

the network control information comprises: network access control information or network connection overhead bit fields; and

the handling comprises: forming, inserting, transmitting, receiving, connecting or applying.

9. A network management system comprising:

a file server configured to store binary network element (NE) control files;

a plurality of NEs in communication with the file server;

wherein a set of repeating automatic file transfer routines is configured to transfer the NE control files from the file server to the NEs, with the NE control files representing contents of segments within memory spaces of the NEs associated therewith, and whereby a given one of the NEs copies its respective NE control files from the NMS server to a control register segment within its memory spaces;

wherein the given NE comprises digital hardware logic configured to perform a set of NE-side actions for network management operations associated with a given one of its respective NE control files, with such NE-side actions performed i) based on binary contents of the given NE control file indicating a set of intended network management actions to be completed and ii) automatically by hardware without software involvement, at least when the contents of the given NE control file does not contain a value that causes software to reboot the NE;

wherein said NE-side actions for network management operations associated with the given NE control file comprise configuring, through contents of the control register segment within the NE memory spaces, how the NE hardware, automatically without software involvement, is to function, wherein said contents of the control register segment are direct copies of the binary contents of the given NE control file; and

wherein the NE-side actions performed automatically by hardware without software involvement include producing a set of NE status parameters, with such producing comprising: forming NE alarm indicators based at least in part on NE defect bits and alarm enable bits, wherein values of the alarm enable bits are based on the contents of the given NE control file, and wherein the set of NE status parameters include: a defect indicator, an intermediary level alarm indicator, or a top level alarm indicator for the NE.

10. The system of claim **9** wherein the given NE is configured to copy the given NE control file from its respective directory at the file server designated for control files for that given NE.

11. The system of claim **9** wherein the set of repeating automatic file transfer routines are further configured to transfer NE program files from the file server to program memory segments of the NEs.

12. The system of claim **9** wherein the given NE is a separate and remote node in a network from the perspective of the file server.

13. The system of claim **9** wherein the NE-side actions performed automatically by hardware without software involvement include: connecting network traffic in an alternative manner, at least in part based on control bits associated with such connecting, wherein values of said control bits are based on the contents of the given NE control file.

14. The system of claim **13** wherein the connecting comprises: multiplexing, redirecting or re-forwarding.

15. The system of claim **9** wherein the NE-side actions performed automatically by hardware without software involvement include: handling network control information in one of alternative manners, at least in part based on control bits associated with such handling, wherein values of said control bits are based on the contents of the given NE control file.

16. The system of claim **15** wherein:

the network control information comprises: network access control information or network connection overhead bit fields; and

the handling comprises: forming, inserting, transmitting, receiving, connecting or applying.

* * * * *